



ILS-Z - 511 Database Design

Fitness Tracking System

Group Name: Query Titans

Final Project Report

By:

**Vaishnavi Pawar
Sreekavya Kashamshetty
Vaishnavi Rai**

April 24, 2024

Database Scenario: FitTrackPro - A Comprehensive Fitness Tracking Platform

Introduction:

The proposal is being discussed for a startup called "FitTrackPro," which focuses on providing digital health and fitness solutions. FitTrackPro is a dynamic startup at the intersection of technology and wellness, created to serve the ever-increasing health-conscious population. FitTrackPro is at the forefront of the health technology industry, innovatively merging fitness monitoring with full health management solutions to appeal to today's health-conscious individuals. FitTrackPro, founded on the premise of boosting personal wellness through technology, provides a powerful digital platform that enables users to actively track a variety of health parameters, including physical activity and nutritional consumption. Our goal is to empower consumers to make informed health decisions by leveraging real-time data and personalized insights to promote a better lifestyle.

FitTrackPro, a pioneer in the digital health business, is dedicated to changing how individuals connect with their health regimens. The firm is intended to serve a growing generation that values wellness and wants to effortlessly incorporate health management into their daily life. FitTrackPro's program promotes a proactive approach to health by emphasizing user-friendly interfaces and data-driven capabilities, enabling users to actively participate with their fitness and nutritional goals.

Database Scenario:

FitTrackPro's product revolves around its powerful database system, which was precisely designed to manage a wide range of health and activity data. This system facilitates the dynamic gathering and analysis of user-generated health information, allowing for tailored feedback and trend analysis, which are critical to the user experience. The database is more than just a collection of knowledge; it is an important tool in our users' wellness journeys, meant to motivate and educate them as they strive for healthier lifestyles. FitTrackPro's database fosters a supportive and engaged community of individuals dedicated to preserving their health and wellness by incorporating tools such as goal setting, progress monitoring, and community challenges.

FitTrackPro is revolutionizing the confluence of technology and personal health care with this integrated digital approach, making wellness more accessible and enjoyable for people globally. FitTrackPro's database design represents our commitment to scalability and agility, ensuring that as the user base increases and health technology evolves, it remains at the forefront of the industry.

Purpose of this Database:

The FitTrackPro database's primary goal is to act as a multidimensional repository for health and fitness data, allowing users to precisely and easily manage their wellness journey. It is precisely designed to handle extensive user profiles, gathering a wide range of physiological data, personalized training routines, and food planning. This vast database not only tracks individual workout sessions, food intakes, and progress toward predetermined targets, but it also powers an adaptive analytics engine that provides individualized insights and recommendations. By combining such diverse data sources, the database claims to provide a personalized experience that resonates with each user's specific exercise goals and nutritional preferences.

Furthermore, the database is designed to promote community involvement and engagement via a variety of collaborative elements such as group challenges and event participation. It serves as a digital conduit, connecting individuals with similar interests and cultivating a supportive network based on shared experiences and collective accomplishments. The addition of a rewards system to the database design adds a layer of motivation, encouraging users to engage consistently and stay longer. The FitTrackPro database, with its precisely built features, not only improves individual recording and management of fitness activities but also transforms the overall experience into a socially stimulating and collaborative adventure in personal health.

Intended Users of FitTrackPro Database:

Individual Users: FitTrackPro helps fitness enthusiasts, athletes, and anyone trying to live a healthy lifestyle track their progress, set realistic objectives, take part in events, and adhere to customized nutrition regimens.

Fitness Trainers and Coaches: Experts in the field can utilize the platform to oversee and manage several clients, create and modify exercise regimens, assess advancement, and offer tailored advice based on real-time data.

FitTrackPro Analysts and Developers: Staff members who will have access to the database to monitor app performance, analyze user data for insights, and create new features.

Healthcare Professionals: With user consent, healthcare providers may utilize the data in specific situations to improve fitness and wellness-related healthcare services.

Event Organizers: The platform can be utilized by individuals or groups to oversee event details, participant registrations, performance tracking, and award distribution for fitness events, challenges, or competitions.

Dietitians and nutritionists: These professionals in health and nutrition can work with clients to create and oversee individualized nutrition plans that will guarantee clients satisfy their dietary needs and attain the best possible health results.

The database has the following entities and relationships:

Entities and attributes:

1. User Entity: Represents the individuals using the application.

- **Attributes:** UserID (unique), FirstName, LastName, DateOfBirth, Email (unique), PhoneNo. (multi-valued), Height, Weight, Age(derived from DOB), Gender.

Certified User Entity (Subclass), Normal User Entity(Subclass) for User(SuperClass)

Certified User Entity (Subclass)

-**Attributes:** UserID(unique), CertificationLevel.

Normal User Entity(Subclass)

-**Attributes:** UserID(unique), SignupDate.

Here, Certified and Normal users are subclasses of the User superclass, forming an overlapping subclass specialization. This design allows for the unique attributes and relationships associated with certified and normal users to be captured within the overarching user profile. This is overlapping because a user can be either a normal user or a certified user.

2. Activity Entity: Represents the various fitness activities users can record.

- **Attributes:** ActivityID (unique), ActivityType, Duration, CaloriesBurnt, ActivityDate.

3. Goal Entity: Represents the fitness goals set by users.

- **Attributes:** GoalID (unique), GoalDescription, StartDate, EndDate, Status.

4. Tracking entity (Weak Entity): Represents the various tracking activities users can record. This is a weak entity.

-**Attributes:** TrackingID(unique), TrackingWeight, Chest, Waist, Biceps, TrackingDate, Thigh, Hips

The "Tracking" entity is considered weak because it records specific measurements (weight, chest, waist, bicep, thigh, hips) and dates for users. These tracking records don't have a meaningful existence without being associated with a user. For instance, a tracking record of weight or waist measurement only makes sense when tied to the user it belongs to.

While "Tracking" has a TrackingID, the true uniqueness of a tracking record is not just by this ID but in combination with the UserID. The TrackingID might not be unique across different users without the context of which user it belongs.

The "Tracking" entity is owned by the "User" entity. Every tracking record is a property of a specific user, and its lifecycle is dependent on the user. If a user is removed from the database, their tracking records would also logically be removed.

5. Event Entity: Represents special events or challenges hosted on the platform.

- **Attributes:** EventID (unique), EventName, StartDate, EndDate of the event.

6. Reward Entity: Represents rewards or achievements users can earn.

- **Attributes:** RewardID (unique), RewardName, Achieved.

7. Nutrition Entity: Represents nutrition plans associated with users.

Attributes: NutritionPlanID (unique), PlanName, Description, CalorieIntake.

Relationships between the entities:

Here's an expanded overview of potential relationships that we considered among the entities mentioned above in the FitTrackPro database:

1. User - Activity Relationship (Many-to-many): One user can engage in multiple activities, and each activity can be recorded by multiple users. This relationship captures the various fitness activities a user performs.

2. User - Goal Relationship (One-to-many): Each user can set multiple fitness goals, but each goal is specific to one user. This relationship helps in tracking the progress of individual goals for each user.

3. User - Tracking Relationship (One-to-many): This is an identifying relation since Tracking is a weak entity. Each user can have multiple tracking records, capturing various body measurements over time. This is crucial for monitoring physical changes and progress.

4. User - Event Relationship (Many-to-many): Users can participate in multiple events, and each event can have multiple participants. This relationship enhances community engagement and motivates users through communal activities and challenges.

5. User - Reward Relationship (Many-to-many): Users can earn multiple rewards, and each reward can be earned by multiple users. This relationship is key for gamification and motivation, encouraging users to achieve specific milestones or participate in events.

6. User - Nutrition Plan Relationship (One-to-many): A user can follow multiple nutrition plans, but each plan is tailored to an individual user. This relationship supports personalized nutrition guidance and tracking.

7. Event - Reward Relationship (One-to-many): Participation or performance in an event can be linked to one or more rewards. This incentivizes participation and achievement in events.

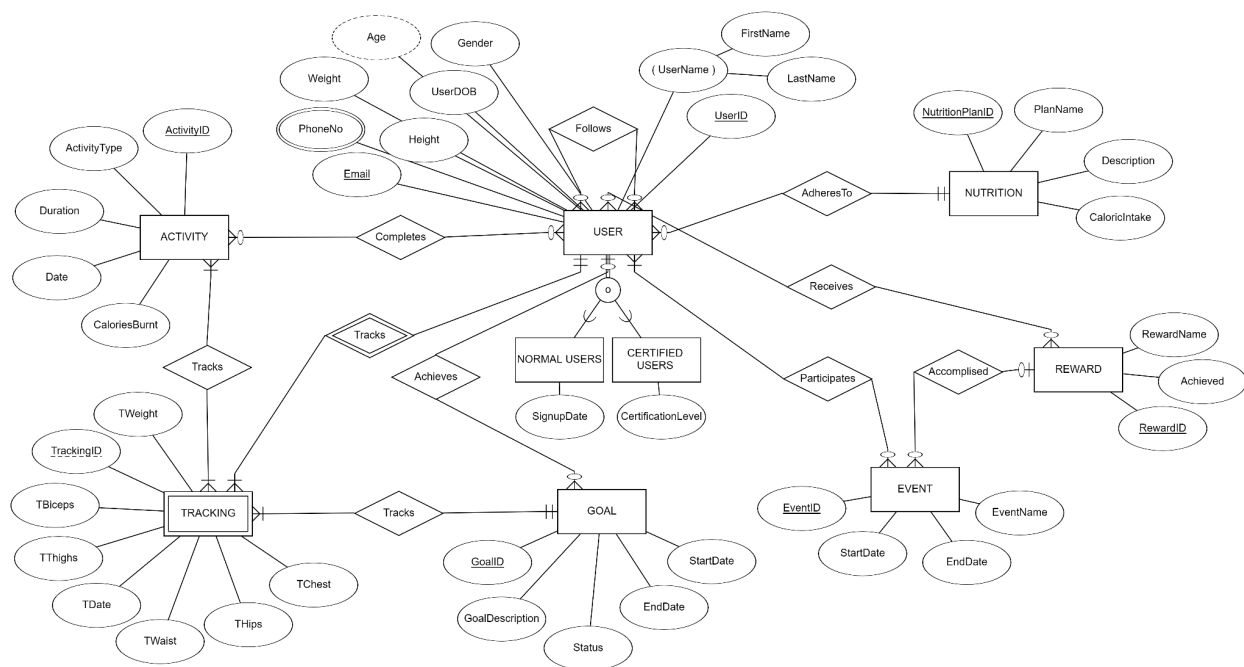
8. Tracking - Goal Relationship (One-to-many): Tracking records can be associated with specific goals, especially if the goals are related to physical measurements. This relationship helps in analyzing the progress toward goal attainment based on tracked measurements.

9. User - User Recursive Relationship: This could represent a following or friendship system, where users can connect with other users, creating a social network within the app. This relationship can be one-to-many, where a user follows many users but is also followed by many.

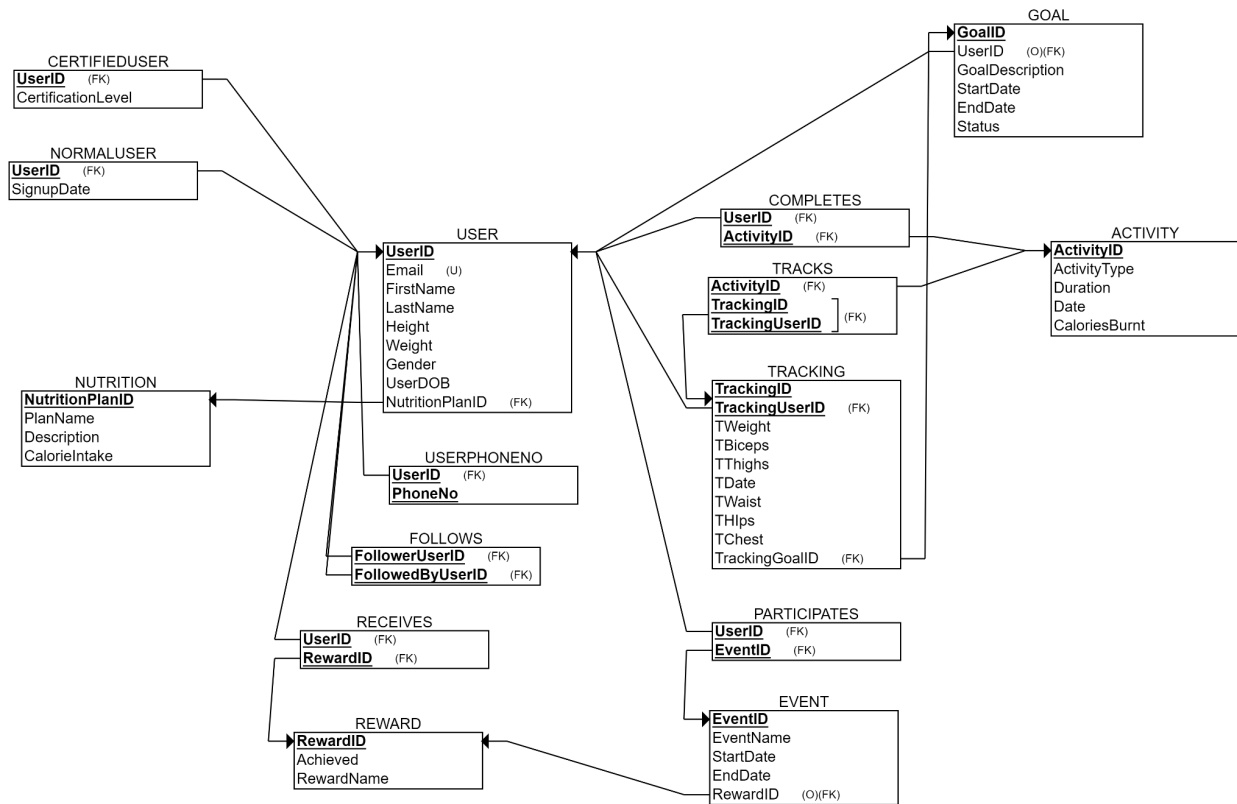
10. Activity - Tracking Relationship (Many-to-many): This could represent the correlation between specific activities and changes in body measurements. For example, tracking the impact of certain activities on weight or other body dimensions over time.

These above relationships, along with the entities and their attributes, form a comprehensive model that supports various features of the FitTrackPro application, from activity tracking and goal setting to event participation and nutrition planning.

Entity Relational Diagram for Fitness Tracking Database:



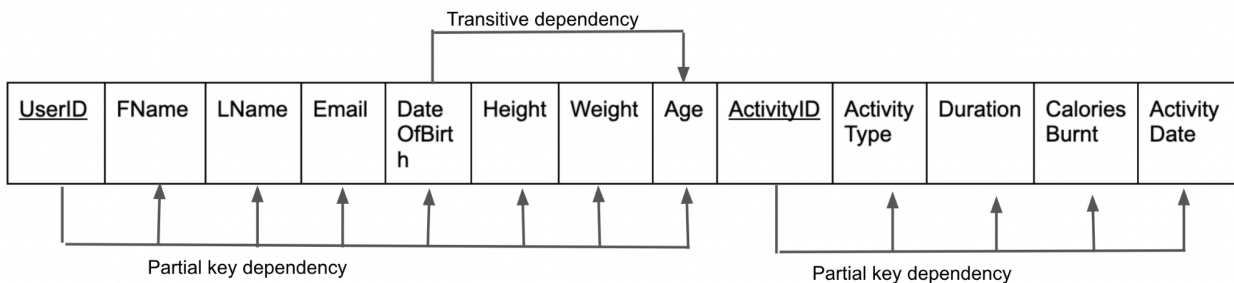
Relational Schema for Fitness Tracking Database:



Denormalization:

Denormalizing the Activity table by adding User-related information. This gives the User-Activity Table as below.

User-Activity table:



Advantages of Denormalization:

1. **Improved Query Performance:** By storing user and activity information in one table, the database can retrieve all relevant data in a single scan without joining multiple tables. This can significantly reduce the query execution time, especially for read-intensive applications.
2. **Reduced Complexity and Join Operations:** Join operations are computationally expensive. This denormalization reduces the need for joins, which can be beneficial in scenarios where we have large databases and join operations might become a performance bottleneck.
3. **Simplified Queries:** Queries become straightforward since all the information is available in one table.
4. **Better for Certain Workloads:** If the application often requires accessing user information together with their activities, having a denormalized table helps the data more readily accessible, which can be advantageous for certain types of workloads, such as reporting and analytics.

Disadvantages of Denormalization:

While denormalization can offer performance benefits, there are trade-offs to be considered. It can lead to data redundancy, increased storage requirements, and potential challenges with maintaining data consistency. It's only recommended to approach denormalization judiciously, by considering the specific needs and usage patterns of the database.

Normalization to 3NF:

There are no full key functional dependencies from the above table. It only has partial dependencies. So starting with 2NF normalization by removing partial dependencies.

Partial functional dependency:

ActivityID -> ActivityType, Duration, CaloriesBurnt, ActivityDate

UserID -> FName, LName, Email, DateOfBirth, Height, Weight, Age

So by creating a new separate Activity Table, it can be turned to 2NF by removing partial dependencies.

Transitive dependency:

DateOfBirth -> Age

Here the non-key attribute DOB uniquely determines a non-key attribute Age. Normalizing the above-denormalized tables into 3NF by removing the transitive dependency:

User Table:

<u>UserID</u>	<u>ActivityID</u>	FName	LName	Email	DateOfBirth	Height	Weight
---------------	-------------------	-------	-------	-------	-------------	--------	--------

Activity Table:

<u>ActivityID</u>	ActivityType	Duration	CaloriesBurnt	ActivityDate
-------------------	--------------	----------	---------------	--------------

Age Table:

<u>DateOfBirth</u>	Age
--------------------	-----

Now all the above 3 tables are in 3NF Normalized form.

Reasons to Maintain Normalized Tables

1. Data Integrity and Redundancy

- Concern: Denormalization can cause data redundancy and inconsistency, especially when data is updated often.
- Justification: Normalization ensures that each data element is only saved once, resulting in a single source of truth and data consistency.

2. Complexity in Handling Data Updates

- Concern: Denormalization complicates data updates by needing changes across numerous tables.
- Justification: Normalization simplifies data management, making changes easier and lowering the possibility of errors.

3. Scalability and Storage

- Concern: Duplicated data causes denormalized tables to grow in size, increasing storage costs and potentially lowering system performance as scale grows.
- Justification: Normalized tables eliminate excessive data duplication, improving scalability and making the database more comprehensible.

Data Source

During our meeting to generate sample data for our Fitness Tracking System, we combined creative inspiration with a variety of credible sources. Our team's imaginative ideas were bolstered by real-world data, resulting in a dataset that reflects the complexity and diversity of actual fitness behaviors.

We recognize the value of authenticity and have included elements of genuine athletic endeavors in our user profiles. Our activity and goal data reflect common fitness benchmarks while allowing for unique personal experiences, emulating the mix of uniformity and individuality found in real-life workouts.

To create plausible locations and user details, we used resources known for their accuracy and depth:

<https://health.gov/our-work/nutrition-physical-activity/dietary-guidelines>

<https://www.myfitnesspal.com/>

<https://www.healthifyme.com/us/>

<https://www.excellenceinfitness.com/blog/nutrition-for-fitness-training-the-facts-you-need-to-know>

Our approach to gathering this data has been both practical and innovative, ensuring that our database design remains true to form. For example, nutritional data was created using information from the above nutrition website. Our resulting database, despite being filled with fictionalized data, meets the functional requirements of a comprehensive fitness tracking system. It has been designed to effectively simulate operational scenarios found in such systems, ranging from tracking routine exercise activities to the complex relationships between diet, fitness events, and reward achievements.

Through this process, we created a dataset that not only serves as a reliable testing platform, but also captures the essence of the fitness industry: dynamic, detailed, and data-driven.

Data Types

We unanimously decided to use fixed-length CHAR data types for all identification keys across the tables when building our Fitness Tracking System database, to promote clarity and consistency. Each identifier consists of a single leading character that represents the entity type, followed by a five-digit number to ensure its uniqueness (for example, 'U' for User in U10001, 'A' for Activity in A10001).

Specifically tailored identification cases include:

UserID: CHAR(6), beginning with 'U'.

ActivityID: CHAR(6), with 'A' prefix.

GoalID: CHAR(6), preceded by 'G'.

TrackingID: CHAR(6), as introduced by 'T'.

EventID is CHAR(6), starting with 'E'.

The RewardID is CHAR(6), prefixed with 'R'.

NutritionPlanID is CHAR(6), beginning with 'N'.

We used the ENUM data type for attributes that store key status and categorical data within entities, such as goal status or activity type. This strategic decision serves to limit the values within a predefined set, reducing the risk of data inconsistencies and increasing report efficiency.

When it came to storage allocation and data precision, we chose INT for integral numbers and DECIMAL for measurements that required fractional precision. For example, attributes like Height, Weight, and Age in the User entity are marked as INT, whereas precise body measurements in the Tracking entity, such as chest, waist, bicep, thigh, and hips are marked as DECIMAL(5,2).

We use the DATE data type for Event and Goal entities, which require accurate date-time stamps for StartDate and EndDate. This ensures that temporal data is recorded uniformly throughout the system.

The deliberate selection of these data types ensures data integrity and streamlines database interactions, paving the way for clear data retrieval and reporting processes.

Q. Why did you add the referential integrity constraints?

Referential integrity constraints are used to assure data consistency and accuracy by enforcing table relationships, ensuring that all references are legitimate and that foreign keys correctly point to existing primary keys. These restrictions avoid orphan entries while also maintaining the database's logical integrity, which is critical for reliable and robust application behavior.

SQL Queries Performed:

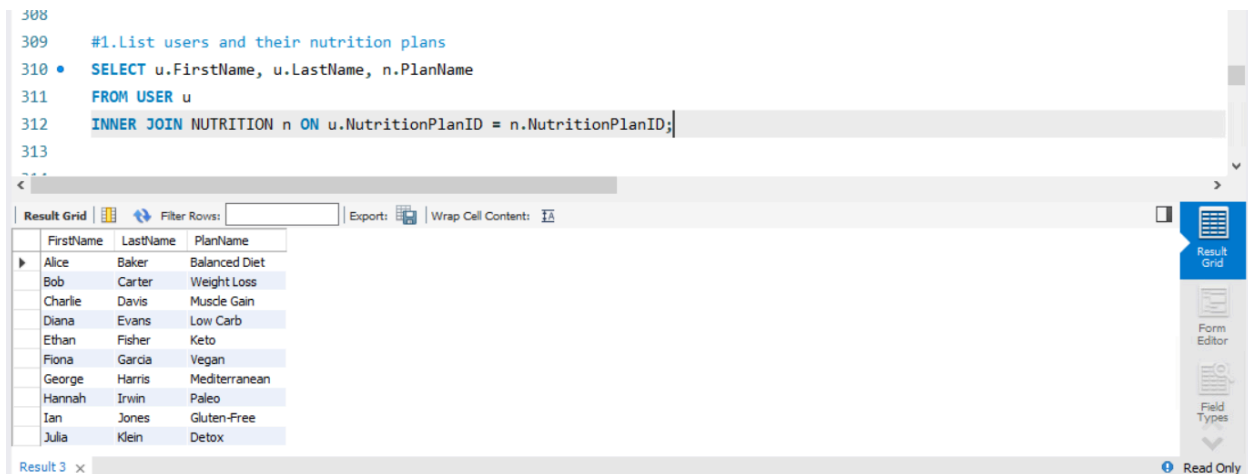
Query 1: List users and their nutrition plans

- Purpose: Retrieves each user's name and associated nutrition plan. Useful for nutritional oversight and personalized dietary recommendations.
- Explanation: This query uses an inner join between the USER and NUTRITION tables to connect users with their specific nutrition plans based on the NutritionPlanID.

```
SELECT u.FirstName, u.LastName, n.PlanName
```

```
FROM USER u
```

```
INNER JOIN NUTRITION n ON u.NutritionPlanID = n.NutritionPlanID;
```



```
308
309 #1.List users and their nutrition plans
310 • SELECT u.FirstName, u.LastName, n.PlanName
311 FROM USER u
312 INNER JOIN NUTRITION n ON u.NutritionPlanID = n.NutritionPlanID;
313
```

FirstName	LastName	PlanName
Alice	Baker	Balanced Diet
Bob	Carter	Weight Loss
Charlie	Davis	Muscle Gain
Diana	Evans	Low Carb
Ethan	Fisher	Keto
Fiona	Garcia	Vegan
George	Harris	Mediterranean
Hannah	Irwin	Paleo
Ian	Jones	Gluten-Free
Julia	Klein	Detox

Result 3 x Read Only

Query 2. Count the number of users per gender

- Purpose: Provides a demographic breakdown of users by gender, useful for market analysis and tailoring gender-specific programs.
- Explanation: Groups users by gender and counts the number in each group, leveraging a simple aggregate function.

```
SELECT Gender, COUNT(*) AS TotalUsers
```

```
FROM USER
```

```
GROUP BY Gender;
```

```

329 # 2. Count the number of users per gender
330 • SELECT Gender, COUNT(*) AS TotalUsers
331 FROM USER
332 GROUP BY Gender;
333
334

```

Gender	TotalUsers
Male	5
Female	5

Query 3: Classify users based on their weight

- Purpose: Categorizes users into weight classes, facilitating targeted health and fitness programs.
- Explanation: Uses a CASE statement to assign a weight category based on the Weight column.

```

SELECT UserID, Weight,
CASE
    WHEN Weight < 50 THEN 'Underweight'
    WHEN Weight BETWEEN 50 AND 70 THEN 'Normal'
    WHEN Weight > 70 THEN 'Overweight'
END AS WeightCategory
FROM USER;

```

```

334 #3Classify users based on their weight
335 • SELECT UserID, Weight,
336 CASE
337     WHEN Weight < 50 THEN 'Underweight'
338     WHEN Weight BETWEEN 50 AND 70 THEN 'Normal'
339     WHEN Weight > 70 THEN 'Overweight'
340 END AS WeightCategory
341 FROM USER;

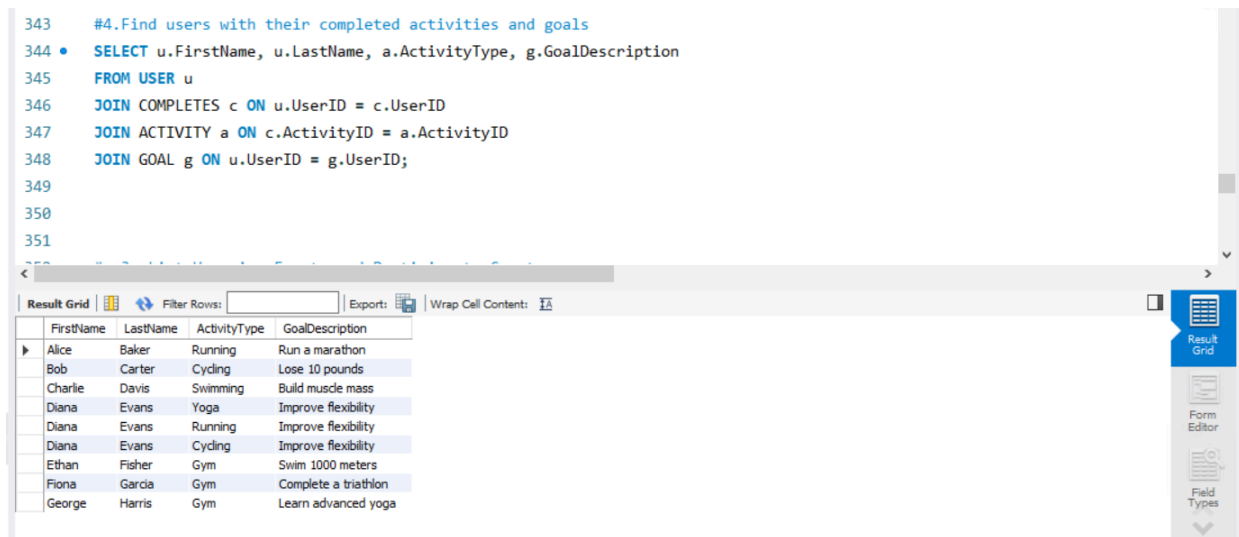
```

UserID	Weight	WeightCategory
U00001	65	Normal
U00002	88	Overweight
U00003	70	Normal
U00004	55	Normal
U00005	80	Overweight
U00006	62	Normal
U00007	76	Overweight
U00008	58	Normal
U00009	90	Overweight
U00010	61	Normal

Query 4: Find users with their completed activities and goals

- Purpose: Links users to their specific activities and goals to review progress and completion rates.
- Explanation: Combines multiple joins to connect users with their activities and goals, giving a comprehensive view of individual achievements.

```
SELECT u.FirstName, u.LastName, a.ActivityType, g.GoalDescription
FROM USER u
JOIN COMPLETES c ON u.UserID = c.UserID
JOIN ACTIVITY a ON c.ActivityID = a.ActivityID
JOIN GOAL g ON u.UserID = g.UserID;
```



The screenshot shows a database query editor with the following SQL code:

```
343 #4.Find users with their completed activities and goals
344 • SELECT u.FirstName, u.LastName, a.ActivityType, g.GoalDescription
345 FROM USER u
346 JOIN COMPLETES c ON u.UserID = c.UserID
347 JOIN ACTIVITY a ON c.ActivityID = a.ActivityID
348 JOIN GOAL g ON u.UserID = g.UserID;
349
350
351
```

Below the code, the 'Result Grid' shows the following data:

FirstName	LastName	ActivityType	GoalDescription
Alice	Baker	Running	Run a marathon
Bob	Carter	Cycling	Lose 10 pounds
Charlie	Davis	Swimming	Build muscle mass
Diana	Evans	Yoga	Improve flexibility
Diana	Evans	Running	Improve flexibility
Diana	Evans	Cycling	Improve flexibility
Ethan	Fisher	Gym	Swim 1000 meters
Fiona	Garcia	Gym	Complete a triathlon
George	Harris	Gym	Learn advanced yoga

Query 5: Total calories burnt by each user

- Purpose: Summarizes the total calories each user has burnt through activities, crucial for tracking fitness outcomes.
- Explanation: Joins the USER, COMPLETES, and ACTIVITY tables and sums calories burnt, grouped by user.

```
SELECT u.FirstName, u.LastName, SUM(a.CaloriesBurnt) AS TotalCaloriesBurnt
FROM USER u
JOIN COMPLETES c ON u.UserID = c.UserID
JOIN ACTIVITY a ON c.ActivityID = a.ActivityID
GROUP BY u.UserID;
```

```

351 #5.Total calories burnt by each user
352 • SELECT u.FirstName, u.LastName, SUM(a.CaloriesBurnt) AS TotalCaloriesBurnt
353 FROM USER u
354 JOIN COMPLETES c ON u.UserID = c.UserID
355 JOIN ACTIVITY a ON c.ActivityID = a.ActivityID
356 GROUP BY u.UserID;
357

```

FirstName	LastName	TotalCaloriesBurnt
Alice	Baker	280
Bob	Carter	300
Charlie	Davis	250
Diana	Evans	820
Ethan	Fisher	450
Fiona	Garcia	450
George	Harris	450

Result 8 x Read Only

Query 6: Classify total duration of activities into categories

- Purpose: Categorizes users based on their total activity duration to tailor fitness challenges or rewards.
- Explanation: Sums the duration of activities per user and categorizes the total using a CASE statement.

```

SELECT u.UserID, u.FirstName, u.LastName,
CASE
    WHEN SUM(a.Duration) < 100 THEN 'Light Exerciser'
    WHEN SUM(a.Duration) BETWEEN 100 AND 300 THEN 'Moderate Exerciser'
    WHEN SUM(a.Duration) > 300 THEN 'Heavy Exerciser'
END AS ExerciseLevel
FROM USER u
JOIN COMPLETES c ON u.UserID = c.UserID
JOIN ACTIVITY a ON c.ActivityID = a.ActivityID
GROUP BY u.UserID;

```

```

358 #6 Classify total duration of activities into categories
359 • SELECT u.UserID, u.FirstName, u.LastName,
360 CASE
361 WHEN SUM(a.Duration) < 100 THEN 'Light Exerciser'
362 WHEN SUM(a.Duration) BETWEEN 100 AND 300 THEN 'Moderate Exerciser'
363 WHEN SUM(a.Duration) > 300 THEN 'Heavy Exerciser'
364 END AS ExerciseLevel
365 FROM USER u
366 JOIN COMPLETES c ON u.UserID = c.UserID
367 JOIN ACTIVITY a ON c.ActivityID = a.ActivityID
368 GROUP BY u.UserID;

```

UserID	FirstName	LastName	ExerciseLevel
U00001	Alice	Baker	Light Exerciser
U00002	Bob	Carter	Light Exerciser
U00003	Charlie	Davis	Light Exerciser
U00004	Diana	Evans	Moderate Exerciser
U00005	Ethan	Fisher	Light Exerciser
U00006	Fiona	Garcia	Light Exerciser
U00007	George	Harris	Light Exerciser

Query 7: List of Users Participating in Each Event

- Purpose: Helps event organizers monitor attendance and manage logistics for each event.
- Explanation: Uses joins to link the EVENT, PARTICIPATES, and USER tables, displaying a sorted list of participants for each event to facilitate organization and communication.

```

SELECT e.EventName, u.FirstName, u.LastName
FROM EVENT e
JOIN PARTICIPATES p ON e.EventID = p.EventID
JOIN USER u ON p.UserID = u.UserID
ORDER BY e.EventName, u.LastName;

```

```

368 #7 List of Users Participating in Each Event
369 • SELECT e.EventName, u.FirstName, u.LastName
370 FROM EVENT e
371 JOIN PARTICIPATES p ON e.EventID = p.EventID
372 JOIN USER u ON p.UserID = u.UserID
373 ORDER BY e.EventName, u.LastName;
374

```

EventName	FirstName	LastName
City Marathon	Alice	Baker
Countryside Bike Rally	Bob	Carter
Countryside Bike Rally	Charlie	Davis
Countryside Bike Rally	George	Harris
Countryside Bike Rally	Ian	Jones
Meditation Retreat	Bob	Carter
Meditation Retreat	Diana	Evans
Open Water Swim	Bob	Carter
Open Water Swim	Hannah	Irwin
Powerlifting Competition	Ethan	Fisher
Powerlifting Competition	Fiona	Garcia

Query 8: List users, their last activity date and activity status

- Purpose: Provides insights into user activity levels, identifying recently active users for engagement strategies.
- Explanation: Uses a complex join to calculate the last activity date for each user and classifies them as 'Active Recently' or 'Inactive' based on this date.
- Note: Here the Activity status depends on the current date. In our Data we have not added recent dates, hence most of the Activity Status is Inactive

```
SELECT u.UserID, u.FirstName, u.LastName, MAX(a.ActivityDate) AS  
LastActivityDate,
```

```
CASE
```

```
    WHEN MAX(a.ActivityDate) >= CURDATE() - INTERVAL 7 DAY THEN 'Active  
Recently'
```

```
    ELSE 'Inactive'
```

```
END AS ActivityStatus
```

```
FROM USER u
```

```
JOIN COMPLETES c ON u.UserID = c.UserID
```

```
JOIN ACTIVITY a ON c.ActivityID = a.ActivityID
```

```
GROUP BY u.UserID;
```

```
377 #8. List users, their last activity date and activity status
378 • SELECT u.UserID, u.FirstName, u.LastName, MAX(a.ActivityDate) AS LastActivityDate,
379 CASE
380     WHEN MAX(a.ActivityDate) >= CURDATE() - INTERVAL 7 DAY THEN 'Active Recently'
381     ELSE 'Inactive'
382 END AS ActivityStatus
383 FROM USER u
384 JOIN COMPLETES c ON u.UserID = c.UserID
385 JOIN ACTIVITY a ON c.ActivityID = a.ActivityID
386 GROUP BY u.UserID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

UserID	FirstName	LastName	LastActivityDate	ActivityStatus
U00001	Alice	Baker	2024-04-01	Inactive
U00002	Bob	Carter	2024-04-02	Inactive
U00003	Charlie	Davis	2024-04-03	Inactive
U00004	Diana	Evans	2024-04-07	Inactive
U00005	Ethan	Fisher	2024-04-05	Inactive
U00006	Fiona	Garcia	2024-04-05	Inactive
U00007	George	Harris	2024-04-05	Inactive

Result
Grid

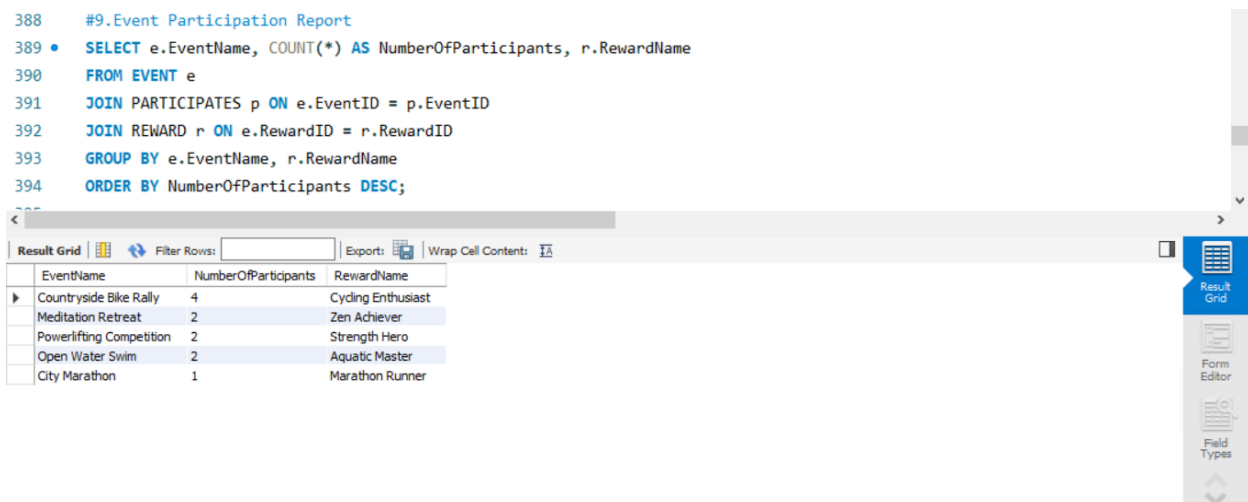
Form
Editor

Field
Types

Query 9: Event Participation Report

- Purpose: Assesses event popularity and the appeal of associated rewards to aid in future planning and marketing strategies.
- Explanation: Groups participants by event and reward, counting the number of participants to evaluate event success and reward effectiveness, highlighting the most popular events.

```
SELECT e.EventName, COUNT(*) AS NumberOfParticipants, r.RewardName
FROM EVENT e
JOIN PARTICIPATES p ON e.EventID = p.EventID
JOIN REWARD r ON e.RewardID = r.RewardID
GROUP BY e.EventName, r.RewardName
ORDER BY NumberOfParticipants DESC;
```



The screenshot shows a database query editor with the following SQL query:

```
388 #9.Event Participation Report
389 • SELECT e.EventName, COUNT(*) AS NumberOfParticipants, r.RewardName
390 FROM EVENT e
391 JOIN PARTICIPATES p ON e.EventID = p.EventID
392 JOIN REWARD r ON e.RewardID = r.RewardID
393 GROUP BY e.EventName, r.RewardName
394 ORDER BY NumberOfParticipants DESC;
```

Below the query, the results are displayed in a table with the following data:

EventName	NumberOfParticipants	RewardName
Countryside Bike Rally	4	Cycling Enthusiast
Meditation Retreat	2	Zen Achiever
Powerlifting Competition	2	Strength Hero
Open Water Swim	2	Aquatic Master
City Marathon	1	Marathon Runner

Query 10: Tracking Progress Report

- Purpose: Monitors users' progress towards their fitness goals, focusing on those nearing their goal deadlines to ensure timely interventions.
- Explanation: Filters for active, impending goals, calculating days remaining to goal deadlines and linking user tracking data to provide actionable insights for users and fitness coaches.

```

SELECT u.FirstName, u.LastName, g.GoalDescription, tr.TWeight AS CurrentWeight,
g.EndDate, (g.EndDate - CURDATE()) AS DaysRemaining
FROM TRACKING tr
JOIN USER u ON tr.TrackingUserID = u.UserID
JOIN GOAL g ON tr.TrackingGoalID = g.GoalID
WHERE g.Status = 'In Progress' AND g.EndDate > CURDATE()
ORDER BY DaysRemaining ASC;

```

396 #10.Tracking Progress Report

397 • SELECT u.FirstName, u.LastName, g.GoalDescription, tr.TWeight AS CurrentWeight, g.EndDate,

398 (g.EndDate - CURDATE()) AS DaysRemaining

399 FROM TRACKING tr

400 JOIN USER u ON tr.TrackingUserID = u.UserID

401 JOIN GOAL g ON tr.TrackingGoalID = g.GoalID

402 WHERE g.Status = 'In Progress' AND g.EndDate > CURDATE()

403 ORDER BY DaysRemaining ASC;

Result Grid

Filter Rows:

Export:

Wrap Cell Contents:

FirstName	LastName	GoalDescription	CurrentWeight	EndDate	DaysRemaining
Bob	Carter	Lose 10 pounds	85	2024-05-01	77
Diana	Evans	Improve flexibility	60	2024-07-01	277
Charlie	Davis	Build muscle mass	75	2024-09-01	477
Alice	Baker	Run a marathon	65	2024-10-01	577

Result 15 x

Read Only

How is it closely related to the purpose and intended users of your database?

This query is strongly related to the FitTrackPro database's goal and intended users, which is to give a detailed and individualized health and fitness monitoring solution to its customers. By calculating the number of unique actions and aggregating activity data, the query directly supports the application's purpose of monitoring and encouraging diverse and consistent fitness routines.

Who will see the report?

The FitTrackPro database generates reports that fitness trainers, dietitians, and health program managers use to modify and improve fitness programs. Furthermore, marketing teams may leverage user interaction data to optimize promotional methods. Finally, the FitTrackPro management team applies these insights to strategic planning and improving the user experience.

What decision will be made based on it?

Decisions based on it:

- The report can guide judgments regarding individualized fitness recommendations to enhance users' routines.
- Providing advertisements for classes or sessions in activities that users have yet to try.
- Recognizing and rewarding users for their various fitness accomplishments.
- Designing challenges or community events to encourage participation in less common activities.

What is the application?

The application would be FitTrackPro's user interface, which users interact with to log activities, track progress, and receive feedback and recommendations based on their fitness activity.

Who are the users?

The primary users in this scenario would be fitness enthusiasts and those who use the FitTrackPro app to track their personal fitness objectives. They may view a dashboard or receive notifications depending on the query results, highlighting their activity diversity and recommending methods to improve their exercise regimen.

Mistakes and their related correction done

1. Incorrect Relationship in Relational Diagram between User and Goal:

- Mistake: The relationship between User and Goal could have been modeled erroneously, possibly as a many-to-many rather than the correct one-to-many relationship.
- Correction: Make sure that each goal is associated with only one user, however one user can have several objectives. This should be explicitly indicated via a foreign key in the Goal table that points back to the User table.

2. Missing Relationships for Activity and Tracking:

- Missing: Important links between Activity and Tracking may have been overlooked. These interactions are critical for understanding how activities influence or correspond with tracked data, such as weight or fitness goals.
- Correction: Defines a many-to-many relationship when many activities affect multiple tracking records and vice versa.

3. Added Attributes in Subclasses (Certified User and Normal User):

- Missing: Attributes in Subclasses were missing, it is typical in database design to add distinguishing attributes that justify their separation from the superclass User.
- Correction: For the subclasses Certified User and Normal User added attributes to distinguish them.

Conclusion

The FitTrackPro database is built to be strong, scalable, and secure, ensuring that it supports the fitness-tracking application's essential features while also providing a personalized and engaging user experience. The database's structure makes it easier to collect, analyze, and show fitness data, which increases user engagement and retention while also giving useful insights to customers and the firm.